Visual-Inertial SLAM

Jay Paek
Department of Electrical and Computer Engineering
University of California, San Diego
La Jolla, California
jpaek@ucsd.edu

***Abstract*—This report presents an individual effort towards implementing visual-inertial simultaneous localization and mapping (SLAM) using an extended Kalman filter (EKF). The project aims to integrate measurements from an inertial measurement unit (IMU) and a stereo camera to estimate the trajectory of the IMU and the positions of visual landmarks in a dynamic environment. The report outlines a systematic approach to address three key tasks: IMU localization via EKF prediction, landmark mapping via EKF update, and the integration of IMU prediction with landmark update to achieve visual-inertial SLAM. A detailed discussion of the problem formulation, technical approach, and results is provided, including insights into successful strategies, challenges encountered, and areas for further improvement. The report also emphasizes the significance of visual-inertial SLAM in robotics and autonomous systems, highlighting its relevance in various real-world applications. Overall, this work contributes to the understanding and implementation of advanced sensor fusion techniques for robust localization and mapping in complex environments.**

***Index Terms*—Robotics, sensor fusion, SLAM, estimation, extended Kalman filter, nonlinear optimization**

I. Introduction

Visual-Inertial Simultaneous Localization and Mapping (SLAM) stands at the forefront of modern robotics and autonomous systems. It addresses the critical need for accurate and robust localization and mapping in dynamic and unstructured environments. As robots navigate through real-world scenarios, they encounter challenges such as varying lighting conditions, dynamic obstacles, and complex geometric structures. Traditional localization and mapping methods often struggle to cope with these challenges, leading to inaccurate or unreliable estimates of robot pose and environment geometry. Visual-Inertial SLAM offers a promising solution by fusing measurements from both visual sensors, such as cameras, and inertial sensors, like accelerometers and gyroscopes, to enhance localization and mapping performance. This integration enables robots to navigate more effectively, enabling applications in fields such as autonomous driving, robotic exploration, and augmented reality.

One fundamental aspect of Visual-Inertial SLAM is IMU localization, which involves estimating the pose of the robot over time using measurements from an Inertial Measurement Unit (IMU). IMUs provide information about the linear and angular velocities of the robot, allowing for the prediction of its trajectory. In this project, we employ an Extended Kalman Filter (EKF) approach to integrate IMU measurements with kinematics equations, enabling accurate pose estimation despite sensor noise and dynamic motion. By leveraging EKF prediction, we aim to track the motion of the robot with high precision, laying the foundation for robust SLAM.

In addition to IMU localization, Visual-Inertial SLAM requires mapping the environment by estimating the positions of visual landmarks observed by the camera. Landmark mapping involves associating detected visual features across multiple frames and estimating their 3D positions in the environment. To achieve this, we implement an EKF-based approach where the unknown landmark positions are treated as states to be estimated. By incorporating visual feature measurements and leveraging EKF update steps, we aim to accurately reconstruct the environment's geometry despite challenges such as occlusions and limited sensor viewpoints.

The core of Visual-Inertial SLAM lies in integrating IMU localization with landmark mapping to achieve a complete SLAM algorithm. By combining the EKF prediction step for IMU localization with the EKF update step for landmark mapping, we create a unified framework for visual-inertial SLAM. This integration enables the system to continuously refine both the robot's pose and the environment's map in a consistent manner. Through careful fusion of IMU and visual measurements, our EKF SLAM algorithm aims to provide accurate and reliable localization and mapping capabilities, even in challenging real-world scenarios.

II. Notations and Preliminaries

In total, we are given measurements indexed by $t = 0, \ldots, T$. The UNIX time stamp for a certain time step will be τ_t and the change in real time between time step t and $t+1$ will be $\Delta\tau_t$. The orientation of the car at times step t will be $T_t \in SE(3) \subset \mathbb{R}^{4\times4}$, where $T_0 = I$, the identity matrix. Any transformation matrix T_t can be decomposed into $R_t \in SO(3)$ and $\mathbf{t}_t \in \mathbb{R}^3$ arranged as follows:

$$T_t = \begin{bmatrix} R_t & \mathbf{t}_t \\ \mathbf{0}^\top & 1 \end{bmatrix}.$$

where

$$SO(3) = \{A \in \mathbb{R}^{3\times3} | A^\top A = I, \det A = 1\}$$

This pose can also be represented as a vector in $\mathbb{R}^6$, and this will be called the axis-angle representation.

Let $\exp(.)$ be defined by the Taylor series expansion of the exponential function. This will allow taking the exponent of matrices.

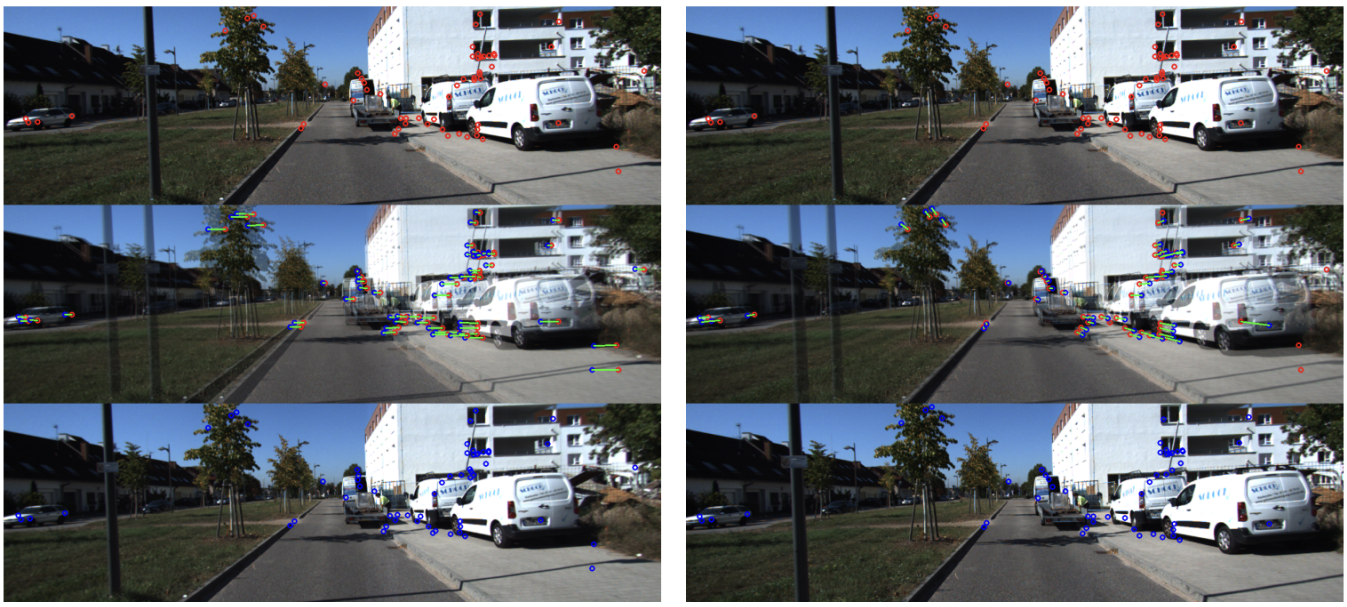

Fig. 1: Visual features matched across the left-right camera frames (left) and across time (right) (Source: ECE276A PR3 doc)

The inertial measurement data have been preprocessed, so the data offers $\mathbf{v}_t, \boldsymbol{\omega}_t \in \mathbb{R}^3$, the linear velocity and angular velocity at time step t. These two vectors are organized as follows:

$$\mathbf{v}_t = \begin{bmatrix} v_x & v_y & v_z \end{bmatrix}^\top, \boldsymbol{\omega}_t = \begin{bmatrix} \omega_x & \omega_y & \omega_z \end{bmatrix}^\top$$

Where v_x, v_y, v_z is the velocities in the x, y, z directions, while $\omega_x, \omega_y, \omega_z$ are the pitch, roll, and yaw values. Combining these two vectors together, we will create the control input vector at time t to be $\mathbf{u}_t \in \mathbb{R}^6$, where $\mathbf{u}_t = \begin{bmatrix} \mathbf{v}_t^\top & \boldsymbol{\omega}_t^\top \end{bmatrix}^\top$.

Let $\mathbf{z}_t \in \mathbb{R}^{4\times M}$ be the pixel measurements of M different features at the time step t. These features have been preprocessed and have been correlated over all times steps. Hence, the jth column of $\mathbf{z}_t$, denoted $\mathbf{z}_{t,j} \in \mathbb{R}^4$ are the pixel measurements of the same feature over all t, formatted as followed:

$$\mathbf{z}_{t,j} = \begin{bmatrix} x_L, y_L, x_R, y_R \end{bmatrix}^T,$$

where x_L, y_L denote the pixel measurement of the feature in the left camera frame, while x_R, y_R denote the pixel measurement of the feature in the right camera frame. If the certain feature has no pixel measurement at a certain time step, then the column will be $\begin{bmatrix} -1, -1, -1, -1 \end{bmatrix}$.

$\mathbf{z}_{t,j}$ will be measuring the landmark $\mathbf{m}_j = \begin{bmatrix} m_x & m_y & m_z \end{bmatrix}^T \in \mathbb{R}^3$, where $m_{j,x}, m_{j,y}, m_{j,z}$ are the world coordinates of the landmark j. Occasionally, we will use $\underline{\mathbf{m}}_j = \begin{bmatrix} \mathbf{m}_j^\top & 1 \end{bmatrix}^T$ to denote homogeneous coordinates.

We are also given parameters to convert information between the camera and the IMU. Let ${}_IT_O$ be the transformation matrix from the optical frame of the camera to the IMU frame, and ${}_OT_I = {}_IT_O^{-1}$ be the transformation in the other direction.

Let K be the intrinsic calibration matrix for a single stereo camera

$$K = \begin{bmatrix} fs_u & 0 & c_u \\ 0 & fs_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$$

which is given. Additionally, the distance between the left and right camera is 0.6 meters in the x-direction of the left camera, so the translation from the left to right camera in the left camera's orientation frame is $\mathbf{p} = \begin{bmatrix} 0.6 & 0 & 0 \end{bmatrix}$. Otherwise there is no orientational difference between the two cameras. With all of this can construct a K_s, the calibration matrix for the stereo camera system:

$$K_s = \begin{bmatrix} fs_u & 0 & c_u & 0 \\ 0 & fs_v & c_v & 0 \\ fs_u & 0 & c_u & -fs_ub \\ 0 & fs_v & c_v & 0 \end{bmatrix}$$

Define the hat map $(.)^\wedge$ (also denoted as $\hat{(.)}$) be a function either from $\mathbb{R}^3 \to \mathbb{R}^{3\times3}$ or $\mathbb{R}^6 \to \mathbb{R}^{4\times4}$. Let $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^\top \in \mathbb{R}^3$ then

$$\hat{\mathbf{x}} = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}.$$

However, if we have a vector like $\mathbf{u}_t \in \mathbb{R}^6$, then

$$\hat{\mathbf{u}}_t = \begin{bmatrix} \hat{\boldsymbol{\omega}}_t & \mathbf{v}_t \\ \mathbf{0}^\top & 1 \end{bmatrix}.$$

Similarly, define the curly hat map $(.)^\curlywedge : \mathbb{R}^6 \to \mathbb{R}^{6\times6}$ where

$$(\mathbf{u}_t)^\curlywedge = \begin{bmatrix} \hat{\boldsymbol{\omega}}_t & \hat{\mathbf{v}}_t \\ \mathbf{0} & \hat{\boldsymbol{\omega}}_t \end{bmatrix}$$

Define the dot map $(.)^{\odot} : \mathbb{R}^4 \to \mathbb{R}^{4\times 6}$ for homogeneous vectors $\underline{\mathbf{v}} = \begin{bmatrix}\mathbf{v} & 1\end{bmatrix}^\top$ where

$$(\underline{\mathbf{v}})^{\odot} = \begin{bmatrix} I & -\hat{\mathbf{v}} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

Define the projection function $\pi : \mathbb{R}^4 \to \mathbb{R}^4$, where if we have $\mathbf{v} = \begin{bmatrix}a & b & c & d\end{bmatrix}^T$, then

$$\pi(\mathbf{v}) = \begin{bmatrix}\frac{a}{c} & \frac{b}{c} & 1 & \frac{d}{c}\end{bmatrix}^T$$

and the vector derivative is the following:

$$\frac{d\pi}{d\mathbf{v}} = \begin{bmatrix} 1 & 0 & -\frac{a}{c} & 0 \\ 0 & 1 & -\frac{b}{c} & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -\frac{d}{c} & 1 \end{bmatrix}$$

Let $\otimes$ be the Kronecker product between two matrices.

Let $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ be a Gaussian normal probability distribution with expected value, or mean, $\boldsymbol{\mu}$ and covariance matrix Σ. The mean and covariance at time step t will be $\boldsymbol{\mu}_t$ and $\boldsymbol{\Sigma}_t$ respectively. Now given information regarding the estimated orientations $\mathbf{x}_t \in \mathbb{R}^6$, control inputs $\mathbf{u}_t$, and observations $\mathbf{z}_t$ over some time steps, we will make the follow simplifications for briefer notation:

$$\mathbf{x}_{t|t} = \mathbf{x}_t | \mathbf{x}_{0:t-1}, \mathbf{u}_{0:t-1}, \mathbf{z}_{0:t}$$

$$\mathbf{x}_{t+1|t} = \mathbf{x}_{t+1} | \mathbf{x}_{0:t}, \mathbf{u}_{0:t-1}, \mathbf{z}_{0:t}$$

And this notation follows for covariance as well.

pdf is used as an acronym for probability distribution function.

III. Problem Formulation

In this paper, we will be using data gathered from a car driving around a town. All of the problems will be solved under the Markov assumption and thus modeled by a Markov chain. All noise presented in each model will be assumed independent.

A. Localization

First, we want to know where our car is at each time step. Let $\mathbf{x}_0$ denote the initial pose, initialized to be the identity pose, in whichever representation we desire to use.

At time t, given the robot's pose $\mathbf{x}_t$ and control input $\mathbf{u}_t$, we can iteratively describe the robot's pose at $t+1$

$$\mathbf{x}_{t+1} = f(\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t)$$

where $\mathbf{w}_t$ is motion noise. Clearly, the distribution of $\mathbf{x}_{t+1}$ is dependent on the information of the previous time step, hence:

$$\mathbf{x}_{t+1} \sim p_f(.|\mathbf{x}_t, \mathbf{u}_t, \mathbf{w}_t).$$

We will call this the motion model. We want to find $\mathbf{x}_t$ such that the following probabilities are maximized:

$$p(\mathbf{x}_t | \mathbf{u}_{0:t-1}), \forall t = 0, \ldots, T$$

Essentially, we are ensuring that the locations of the car obey the velocity and angular velocity measurements that were given.

B. Mapping

Now, under assumption that the $\mathbf{x}_t$ are accurate, we want to map the environment around the map. Let $\mathbf{m}$ be the collection of all $\mathbf{m}_j$. We obtain observations of $\mathbf{m}$, $\mathbf{z}_t$, at each time step. Therefore, there is a dependence between $\mathbf{z}_t$ and $\mathbf{m}$.

$$\mathbf{z}_t = h(\mathbf{x}_t, \mathbf{v}_t)$$

where $\mathbf{v}_t$ is the observation noise. h is considered to be a black box with parameters configured by $\mathbf{m}$. Similar to the motion model, $\mathbf{z}_t$ admits a probability distribution as well.

$$\mathbf{z}_t \sim p_h(.|\mathbf{x}_t, \mathbf{v}_t)$$

We will call this the observation model. We want to find $\mathbf{m}$ such that the following probabilities are maximized:

$$p(\mathbf{m}|\mathbf{x}_t, \mathbf{z}_t), \forall t = 0, \ldots, T$$

Here, we want to make sure that our estimations of the environment align with our observations of the environment from certain orientations.

C. SLAM

Given the motion and observation models, our goal is the estimate the position of our moving body $\mathbf{x}_{0:T}$ and map the environment $\mathbf{m}$ using the control inputs $\mathbf{u}_{0:T}$ and observations $\mathbf{z}_{0:T}$. In more formal terms, we want find $\mathbf{x}_{0:T}$ and $\mathbf{m}$ that maximize the following probabilities

$$p(\mathbf{x}_t, \mathbf{m} | \mathbf{u}_{0:t-1}, \mathbf{z}_{0:t}), \forall t = 0, \ldots, T$$

The parallels between the last two sections and SLAM can definitely be seen. We will approach SLAM problem by first consider the localization problem, then the mapping problem. Finally, we will put those two steps together in order to perform SLAM.

In our specific problem, we have the information $\mathbf{u}_t =$ the linear and angular velocity and $\mathbf{z}_t =$ pixel measurements of $\mathbf{m}_j, j = 1, \ldots, M$ at each time step t. We want to find the best possible estimations of $\mathbf{x}_t$, the positions of the car, and $\mathbf{m}_t$, the positions of the landmarks.

D. Extended Kalman Filter

In this project, we will be using a filtering scheme known as the extended Kalman filter (EKF) where we assume that

- the probability distribution of the all poses and landmarks positions are Gaussian
- $\mathbf{w}_t \sim \mathcal{N}(0, W), \mathbf{v}_t \sim \mathcal{N}(0, V)$.

We use the EKF because we are working with a very complex nonlinear problem, and we want to simplify as many calculations as possible without losing much information. This is done by approximating the nonlinear model as a first-order Taylor polynomial, representing random vectors as their mean, and forcing the posterior pdfs to be Gaussian to allow robust estimation of the pdfs just by the mean and covariance.

Let $\mathbf{x}_{t|t} \sim \mathcal{N}(\mu_{t|t}, \boldsymbol{\Sigma}_{t|t})$ where $\mu_{t|t}, \boldsymbol{\Sigma}_{t|t}$ are given.

First, we obtain the linear approximation of the motion model at the most likely position of $\mathbf{x}_t$ with mostly likely amount of noise to estimate $\mathbf{x}_{t+1|t}$

$$\begin{aligned}\mathbf{x}_{t+1|t} = f(\mathbf{x}_{t|t}, \mathbf{u}_t, \mathbf{w}_t) &\approx f(\boldsymbol{\mu}_{t|t}, \mathbf{u}_t, 0) \\ &+ \frac{\partial f}{\partial \mathbf{x}_{t|t}}\Big|_{\mathbf{x}_{t|t}=\boldsymbol{\mu}_{t|t}, \mathbf{w}_t=0}(\mathbf{x}_{t|t} - \boldsymbol{\mu}_{t|t}) \\ &+ \frac{\partial f}{\partial \mathbf{w}_t}\Big|_{\mathbf{x}_{t|t}=\boldsymbol{\mu}_{t|t}, \mathbf{w}_t=0}(\mathbf{w}_t - 0)\end{aligned}$$

Let $F_t = \frac{\partial f}{\partial \mathbf{x}_{t|t}}\Big|_{\mathbf{x}_{t|t}=\boldsymbol{\mu}_{t|t}, \mathbf{w}_t=0}$ and $Q_t = \frac{\partial f}{\partial \mathbf{w}_t}\Big|_{\mathbf{x}_{t|t}=\boldsymbol{\mu}_{t|t}, \mathbf{w}_t=0}$

Now, using the initial estimate of $\mathbf{x}_{t+1}$ we will linearize the observation model with the same reasoning to estimation $\mathbf{z}_{t+1|t}$

$$\begin{aligned}\mathbf{z}_{t+1|t} = h(\mathbf{x}_{t+1|t}, \mathbf{v}_t) &\approx h(\boldsymbol{\mu}_{t+1|t}, 0) \\ &+ \frac{\partial h}{\partial \mathbf{x}_{t+1|t}}\Big|_{\mathbf{x}_{t+1|t}=\boldsymbol{\mu}_{t+1|t}, \mathbf{v}_t=0}(\mathbf{x}_{t+1|t} - \boldsymbol{\mu}_{t|t}) \\ &+ \frac{\partial h}{\partial \mathbf{v}_t}\Big|_{\mathbf{x}_{t+1|t}=\boldsymbol{\mu}_{t|t}, \mathbf{v}_t=0}(\mathbf{v}_t - 0)\end{aligned}$$

Just like before, let $H_{t+1} = \frac{\partial h}{\partial \mathbf{x}_{t+1|t}}\Big|_{\mathbf{x}_{t+1|t}=\boldsymbol{\mu}_{t+1|t}, \mathbf{v}_t=0}$ and $R_{t+1} = \frac{\partial h}{\partial \mathbf{v}_t}\Big|_{\mathbf{x}_{t+1|t}=\boldsymbol{\mu}_{t|t}, \mathbf{v}_t=0}$.

With these simplifications, the EKF will PREDICT then UPDATE.

In the prediction step, $\mathbf{x}_{t+1|t}$ can't really disobey the motion model. However, since many uncertain entities acted upon each other, the covariance must have an update.

$$\boldsymbol{\mu}_{t+1|t} = f(\boldsymbol{\mu}_{t|t}, \mathbf{u}_t, 0)$$

$$\boldsymbol{\Sigma}_{t+1|t} = F_t \boldsymbol{\Sigma}_{t|t} F_t^\top + Q_t W Q_t^\top$$

and this concludes the prediction step.

Then we must prepare for the update step, where this prediction is corrected based on the observation at $t+1$. We want to make our observation at $t+1$, given our position at $\boldsymbol{\mu}_{t+1|t}$, match the real observation obtained.

$$\boldsymbol{\mu}_{t+1|t+1} = \boldsymbol{\mu}_{t+1|t} + K_{t+1|t}(\mathbf{z}_{t+1} - h(\boldsymbol{\mu}_{t+1|t}, 0))$$

$$\boldsymbol{\Sigma}_{t+1|t+1} = (I - K_{t+1|t} H_{t+1}) \boldsymbol{\Sigma}_{t+1|t}$$

where

$$K_{t+1|t} = \boldsymbol{\Sigma}_{t+1|t} H_{t+1}^\top (H_{t+1} \boldsymbol{\Sigma}_{t+1|t} H_{t+1}^\top + R_{t+1} V R_{t+1}^\top)$$

And this concludes the update step.

IV. Technical Approach

Throughout this paper, we will be applying the EKF in order to estimation the trajectory of the car, as well as the landmarks detected by the stereo cameras. As we applying the estimation method, we will be plotting the trajectory of the means of the distributions.

A. Data Preprocessing

Before using any of the observations $\mathbf{z}_t$, we have to make sure that they are reasonable points. For a stereo camera system, $x_L < x_R$ must be enforced, so for any $\mathbf{z}_{t,j}$ such that $x_L < x_R$, set $\mathbf{z}_{t,j} = \begin{bmatrix}-1 & -1 & -1 & -1\end{bmatrix}$.

Furthermore, in order to same computational cost and use the best data for SLAM, we will make sure to only use features that have more than a certain number of pixel measurements. This can also be tuned, but it can exponentially affect the run time.

B. Localization via Inertial Measurements

We will first initialize $T_0 = I$ and $\boldsymbol{\Sigma}_0$ to our liking. Tuning these parameters will be explained later in this section.

Let $T_{t|t} \sim \mathcal{N}(\boldsymbol{\mu}_{t|t}, \boldsymbol{\Sigma}_{t|t})$ where $\boldsymbol{\mu}_{t|t} \in SE(3), \boldsymbol{\Sigma}_{t|t} \in \mathbb{R}^{6\times 6}$. The covariance matrix has a dimension of 6 because elements in $SE(3)$ have 6 degrees of freedom, so we are keeping track of the xyz coordinates along with its pitch, roll, and yaw orientations instead of each element in $T_{t|t}$. Notice that we are working with a term in $SE(3)$, so we must somehow convert the inertial measurements $\mathbf{u}_t$ to an element of $SE(3)$ in order to apply a well-defined transformation to $T_{t|t}$. We will still have noise distributions as defined in the previous section.

The solution is to apply the hat map to $\mathbf{u}_t \in \mathbb{R}^6$. This allows the angular velocities to be convert to a rotation matrix and linear velocity to a translation vector, all arranged in a transformation matrix. Then, apply the time discretization τ_t and take the exponential map so the the rotation matrix falls into $SO(3)$. Hence, we have the following motion model

$$T_{t+1|t} = f(T_{t|t}, \mathbf{u}_t, \mathbf{w}_t) = T_{t|t} \exp(\tau_t(\mathbf{u}_t + \mathbf{w}_t)^\wedge),$$

and we have the prediction step

$$\boldsymbol{\mu}_{t+1|t} = T_{t|t} \exp(\tau_t \hat{\mathbf{u}}_t)$$

$$\boldsymbol{\Sigma}_{t+1|t} = \exp(-\tau_t(\mathbf{u}_t)^\curlywedge) \boldsymbol{\Sigma}_{t|t} \exp(-\tau_t(\mathbf{u}_t)^\curlywedge)^\top + W$$

since we have $F_t = \exp(-\tau_t(\mathbf{u}_t)^\curlywedge)$ and $Q_t = I$. We are not consider the landmarks yet, so there is not update step.

C. Landmark Mapping

Assume that the $T_{t|t}$s in the previous section are accurate, then we can immediately extract the trajectory via the means of the distributions, which are $\boldsymbol{\mu}_{t+1|t}$. We will now perform the update step exclusively on the landmark $\mathbf{m}_j$ but not the poses.

Before any updates, the distribution of $\mathbf{m}_j$ must be initialized. In order to do so, we will use the first instance of $\mathbf{z}_{t,j}$ that is not $\begin{bmatrix}-1 & -1 & -1 & -1\end{bmatrix}^\top$. Recall that $\mathbf{z}_{t,j} = \begin{bmatrix}x_L & y_L & x_R & y_R\end{bmatrix}^\top$. For simplicity, let $\mathbf{z}_L = \begin{bmatrix}x_L & y_L\end{bmatrix}^\top$ and $\mathbf{z}_R = \begin{bmatrix}x_R & y_R\end{bmatrix}^\top$. The relative rotation from the left camera to the right camera is $R = I$ and the translation is $\mathbf{p}$ as defined in the previous section. For any pixel measurement $\mathbf{z}$, we can obtain the corresponding point in the optical frame $\mathbf{z}'$

$$\underline{\mathbf{z}}' = K^{-1}\underline{\mathbf{z}}$$

Using the measurements in the optical frame, construct $\mathbf{a}, \mathbf{b}$

$$\mathbf{a} = R^\top \mathbf{p} - \mathbf{e}_3^\top R^\top \mathbf{p} \underline{\mathbf{z}}'_R$$

$$\mathbf{b} = R^\top \underline{\mathbf{z}}'_L - \mathbf{e}_3^\top R^\top \underline{\mathbf{z}}'_L \underline{\mathbf{z}}'_R$$

and then we can recover the point in the camera frame

$$\underline{\mathbf{m}}_j = \frac{\mathbf{a}^\top \mathbf{a}}{\mathbf{a}^\top \mathbf{b}} \underline{\mathbf{z}}'_1$$

Due to the simplicity of the camera configurations, we can obtain a better closed form:

$$\mathbf{a} = \mathbf{p}, \mathbf{b} = \underline{\mathbf{z}}'_L - \underline{\mathbf{z}}'_R$$

$$\underline{\mathbf{m}}_j = \frac{0.6}{x'_L - x'_R} \underline{\mathbf{z}}'_L$$

Then initialize $\mathbf{m}_j$ as a probability distribution

$$\mathbf{m}_j \sim \mathcal{N}(g\Big(\frac{0.6}{x'_L - x'_R} \underline{\mathbf{z}}'_L\Big), \mathbf{\Sigma}_j)$$

where g dehomogenizes the homogeneous vector. We can initialize the covariance as desired.

Our observation model is defined by h:

$$h(T_{t+1}, \mathbf{m}_j, \mathbf{v}_t) = K_s \pi({}_O T_I T_{t+1}^{-1} \underline{\mathbf{m}}_j) + \mathbf{v}_t$$

Let $P = \begin{bmatrix} I & \mathbf{0} \end{bmatrix} \in \mathbb{R}^{3\times 4}$, then we have the partial derivative with respect to $\mathbf{m}_j$ with multiply applications of the chain rule

$$\frac{\partial h}{\partial \mathbf{m}_j} = K_s \frac{\partial \pi}{\partial \mathbf{q}} ({}_O T_I T_{t+1}^{-1} \underline{\mathbf{m}}_j) \, {}_O T_I T_{t+1}^{-1} P^T$$

$$\frac{\partial h}{\partial \mathbf{v}_t} = I$$

We want to update all of $\mathbf{m}$ all at once, and we know that they are not necessarily independent. To resolve this, let $\boldsymbol{\mu}_0 = \mathbf{0} \in \mathbb{R}^{3M}$ and $\mathbf{\Sigma} = I_{M\times M} \otimes V$, where M is the number of landmarks that we saw until time t and each mean is stored in triplets of the vector i.e.

$$\mathbf{m} = \begin{bmatrix} \mathbf{m}_1^\top \dots \mathbf{m}_M^\top \end{bmatrix}^\top.$$

We will assume a bijection from observations to landmark. If the jth landmark is first observed, then it will be initialized by the triangulation of the first observation. Afterwards, we will update M.

Let $J_t = \{j = 1, \dots, M | \mathbf{z}_{t,j} \neq \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}^\top\}$.

Construct the vector to store all of the predicted observations at $t+1$ given T_{t+1}.

$$\tilde{\mathbf{z}}_{t+1} = \begin{cases} K_s \pi({}_O T_I T_{t+1}^{-1} \underline{\mathbf{m}}_j) & \text{if } j \in J \\ \begin{bmatrix} -1 & -1 & -1 & -1 \end{bmatrix}^\top & \text{else} \end{cases}$$

Then we construct $H_{t+1} \in \mathbb{R}^{4M\times 3M}$

$$[H_{t+1}]_{4j-4:4j, 3j-3:3j} = \begin{cases} K_s \frac{\partial \pi}{\partial \mathbf{q}} ({}_O T_I T_{t+1}^{-1} \underline{\mathbf{m}}_j) \, {}_O T_I T_{t+1}^{-1} P^T & \text{if } j \in J \\ \mathbf{0} & \text{else} \end{cases}$$

$$\boldsymbol{\mu}_{t+2|t+1} = \boldsymbol{\mu}_{t+1|t} + K_{t+1|t}(\mathbf{z}_{t+1} - \tilde{\mathbf{z}}_{t+1})$$

$$\mathbf{\Sigma}_{t+2|t+1} = (I - K_{t+1|t} H_{t+1}) \mathbf{\Sigma}_{t+1|t}$$

where

$$K_{t+1|t} = \mathbf{\Sigma}_{t+1|t} H_{t+1}^\top (H_{t+1} \mathbf{\Sigma}_{t+1|t} H_{t+1}^\top + I \otimes V)$$

since $R_{t+1} = I_{M\times M} \otimes I = I_{3M\times 3M}$. Thankfully, if a landmark j had just been initialized, then $\mathbf{z}_j - \tilde{\mathbf{z}_j} \approx \mathbf{0}$, so there is no need for additionally filtering.

D. Visual Inertial SLAM

We will now put the two steps together to perform visual inertial SLAM. Since the position of the car and the location of the landmarks can not be assumed to be independent, we must construct a mean vector and covariance matrix that encodes both the pose and the landmarks positions. We will let the noise distribution defined identically as before.

Let $\boldsymbol{\mu}_0 \in \mathbf{0}$ and let $\mathbf{\Sigma}_0$ be chosen as desired.

Let $\boldsymbol{\mu}_{t|t} \in \mathbb{R}^{3M+6}, \mathbf{\Sigma}_{t|t} \in \mathbb{R}^{3M+6\times 3M+6}$, where the $[\boldsymbol{\mu}]_6$, or the first 6 entries of $\boldsymbol{\mu}$, are the axis-angle representation of the pose, while $[\boldsymbol{\mu}]_{3M}$, the last $3M$ entries, are the landmark positions. The covariance matrix will have the following configuration

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{\Sigma}_{PP} & \mathbf{\Sigma}_{PL} \\ \mathbf{\Sigma}_{LP} & \mathbf{\Sigma}_{LL} \end{bmatrix}$$

where $\mathbf{\Sigma}_{PP}$ is the top left 6×6 block that represents the covariance of the pose while the $\mathbf{\Sigma}_{LL}$ is the $3M \times 3M$ block that represents the covariance of the landmarks. $\mathbf{\Sigma}_{PL}$ and $\mathbf{\Sigma}_{LP}$ are of appropriate sizes that represent the cross correlation between the the landmarks and the pose.

Let $\mathbf{y}_{t|t} \sim \mathcal{N}(\boldsymbol{\mu}_{t|t}, \mathbf{\Sigma}_{t|t})$ be our prior distribution. We proceed with the prediction step via the motion model exclusively for $[\mathbf{y}_{t|t}]_6$ as defined in Part B. Let $E = \exp(-\tau_t(\mathbf{u}_t)^\curlywedge)$ then for the covariance update, we do

$$\mathbf{\Sigma}_{t+1|t} = \begin{bmatrix} E\mathbf{\Sigma}_{PP}E^T + W & E\mathbf{\Sigma}_{PL} \\ \mathbf{\Sigma}_{LP}E^T & \mathbf{\Sigma}_{LL} \end{bmatrix}$$

We will obtain $\boldsymbol{\mu}_{t+1|t}, \mathbf{\Sigma}_{t+1|t}$.

Then our update step is similar to Part C. If a landmark has not been initialized, then initialize it using the triangulation method defined in Part C. Since we want to update the pose and the landmarks simulatenously, we have to encode both linearizations into one $H_{t+1} \in \mathbb{R}^{4M\times 3M+6}$ matrix. Separate the matrix into a $4M \times 6$ block for the pose and $4M \times 3M$ block for the landmarks.

The $4M \times 6$ corresponds to the first order Taylor expansion of the observation model with respect to the pose. Let $\boldsymbol{\mu}_{t+1|t} \in \mathbb{R}^6$ be $T_{t+1|t}$. The the partial derivative of h with respect to $T_{t+1|t}$ is the following:

$$\frac{\partial h}{\partial T_{t+1|t}} = K_s \frac{\partial \pi}{\partial \mathbf{q}} ({}_O T_I T_{t+1|t}^{-1} \underline{\mathbf{m}}_j) \, {}_O T_I (T_{t+1|t}^{-1} \underline{\mathbf{m}}_j)^\odot$$

Therefore, for the left $4M + 6$ block,

$$[H_{t+1}]_{4j-4:4j,:6} = \begin{cases} K_s \frac{\partial \pi}{\partial \mathbf{q}}({}_O T_I T_{t+1|t}^{-1} \underline{\mathbf{m}}_j)\, {}_O T_I (T_{t+1|t}^{-1} \underline{\mathbf{m}}_j)^{\odot} & \text{if } j \in J \\ \mathbf{0} & \text{else} \end{cases}$$

We will initialize the $4M \times 3M$ block just as in Part C. Then proceed with the updates with

$$K_{t+1|t} = \boldsymbol{\Sigma}_{t+1|t} H_{t+1}^{\top} (H_{t+1} \boldsymbol{\Sigma}_{t+1|t} H_{t+1}^{\top} + I \otimes V).$$

The landmark updates will be identical to Part C. However, for the mean update

$$\boldsymbol{\mu}_{t+1|t+1} = \boldsymbol{\mu}_{t+1|t} \exp((K_{t+1|t}(\mathbf{z}_{t+1} - \tilde{\mathbf{z}}_{t+1}))^{\wedge})$$

Recall that M is a dynamic value that tracks how many landmarks we have seen so far. This allows the matrices to be smaller at earlier time steps, which makes computation a tad bit faster.

Throughout all of the code, we will be using the compressed sparse row (csr) matrix library from scipy. Due to the sparsity and the large dimensions of the matrices, computing the inverse can be computational heavy and wasteful since most entries are 0. This library allows us to compute inverses and matrix multiplications significiantly faster.

E. Parameter Tuning

It is crucial to select the right parameters when doing SLAM. Higher covariance values means that updates will be small because we're uncertain of the parameter anyway. If our observation error is big, it can be controlled by increasing our initial uncertainty. However, if covariance is low and there is high error in the observations, then the mean is updated drastically. This paper will provide some visualizations on what this looks like.

In this problem, we want to rely on the inertial localization as much as possible. Therefore, we will let $W = 0.0001I$ of the appropriate dimension. Because we do not want the observation to make drastic changes to the trajectory, the observation noise will be set to $V = 100I$ of appropriate dimension.

For the covariance initialization, with similar logic, the position covariance will be initialized to $0.0001I$, while the landmark covariance will be $100I$.

V. Results

For both datasets, the features have been downsampled to around 600 features.

The following plots represent the localization, mapping, and SLAM results for two different datasets. For all of the figure, the landmarks only within an 2-norm less than 1500 from the "center of mass" of the trajectory are shown. One particular note is that when the observation and initial landmark covariance is set to such high values, some landmarks tend to move very far away, to coordinates in the 10000s.

It must be realized that we want to the use the observations as means of correction and not rely on them too much. Throughout parameter tuning, this has been noticed greatly.

This is definitely a trade-off that I willingly took in order to prioritize the localization task over the mapping task. However, this is fine since if we have more features, then some landmarks being incorrect is not too much of a big deal. As long as there are sufficient number of points in relatively correct locations, then our map fulfills its role.

The final figure shows what happens what the observation covariance noises are too small. The incorrect trajectory still definitely resembles the turns in the original trajectory at the right moments. However, these turns are sharper probably due to the feature moving off the camera view very quickly.

VI. Acknowledgements

Thank you to Jay Paek, who worked very hard on this project. Thank you to the professor Atanasov and the wonderful ECE276A TAs for making this hard class enjoyable.

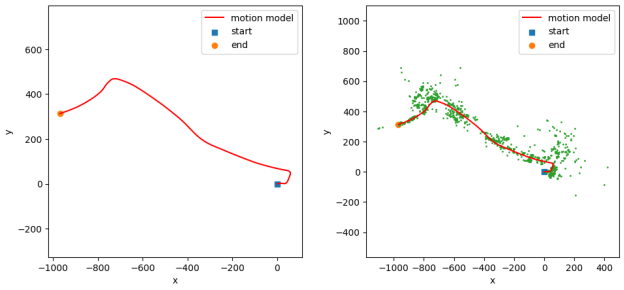

Fig. 2: Motion model (left), initial landmark positions (right) for dataset 03

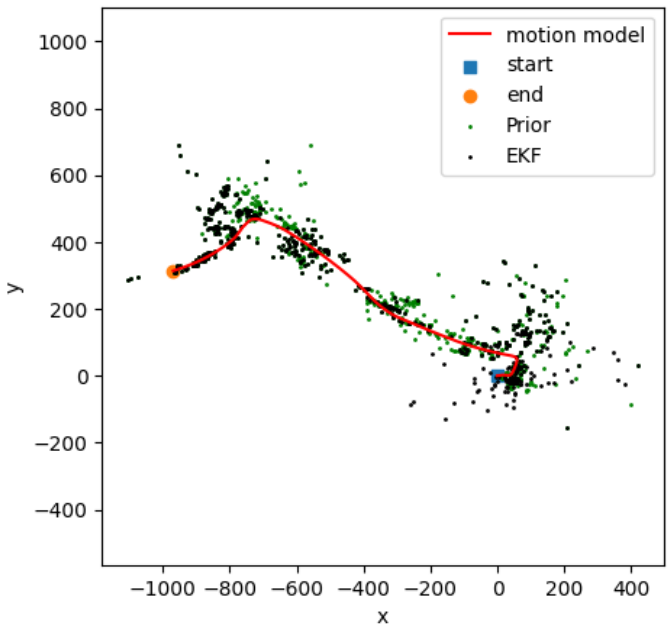

Fig. 3: Motion model (left), initial landmark positions (right) for dataset 03

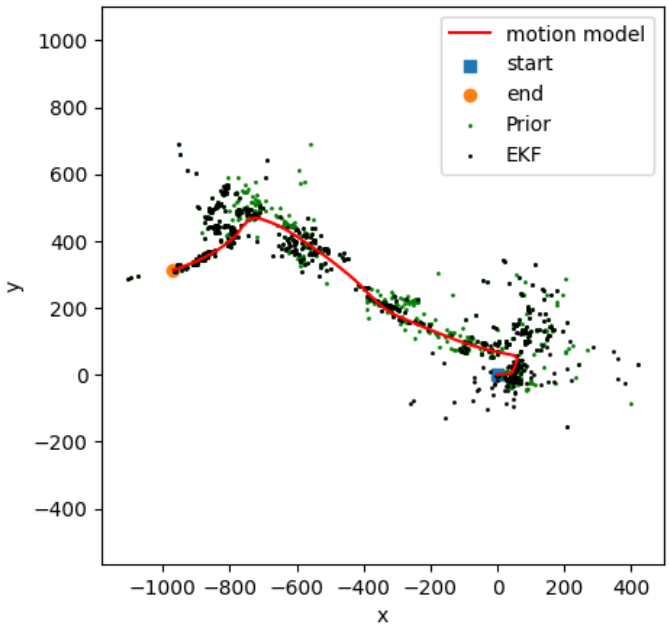

Fig. 4: Prior vs EKF-optimized landmark positions for dataset 03

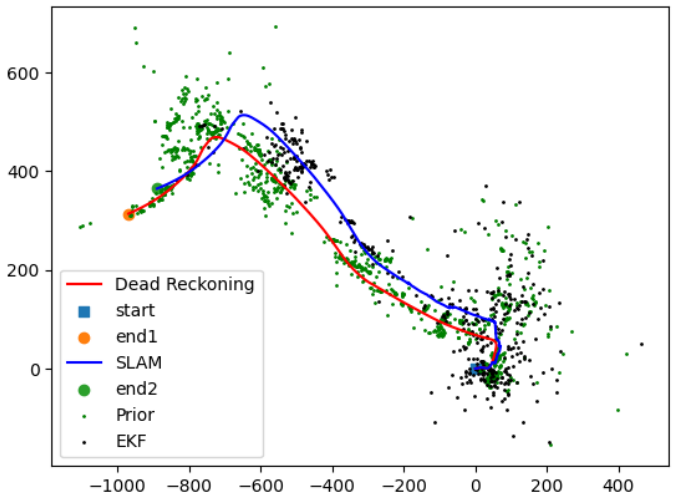

Fig. 5: EKF-SLAM optimized trajectory and landmark positions for dataset 03

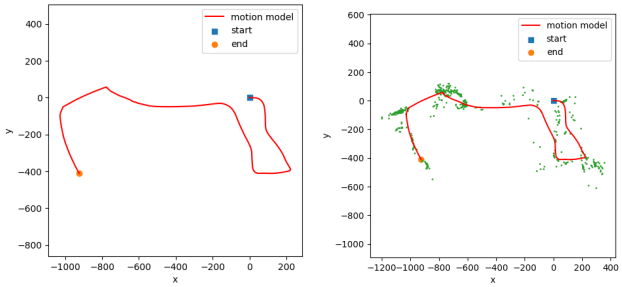

Fig. 6: Motion model (left), initial landmark positions (right) for dataset 10

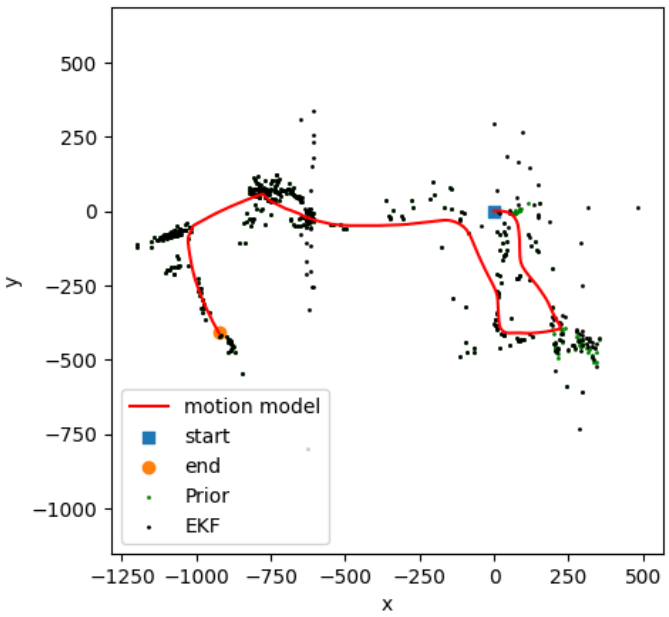

Fig. 7: Prior vs EKF-optimized landmark positions for dataset 10

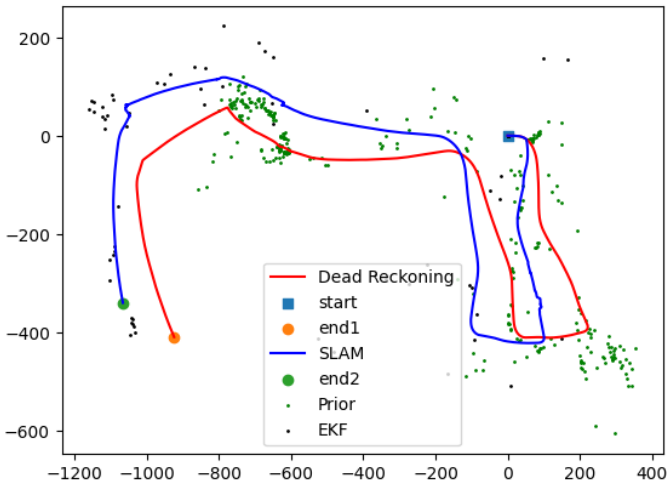

Fig. 8: EKF-SLAM optimized trajectory and landmark positions for dataset 10

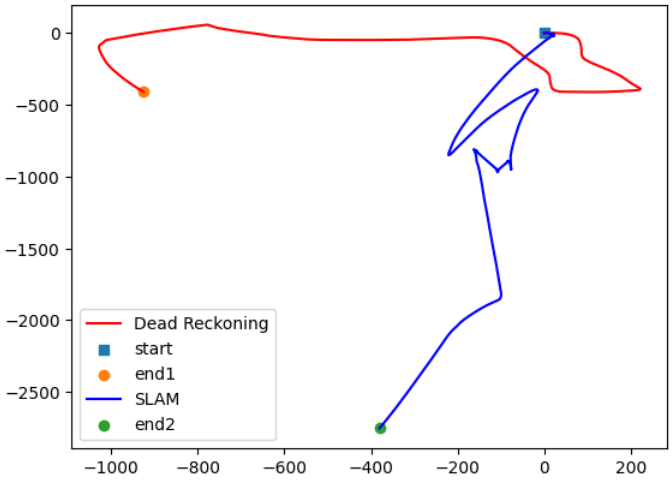

Fig. 9: EKF-SLAM optimized trajectory with poor covariance selection for dataset 10